Research Paper | Computer Science | E-ISSN No : 2454-9916 | Volume : 3 | Issue : 5 | May 2017

GEOSPATIAL EXPAT PARSER USING K-MEANS FOR QUERYING OPEN STREET MAP

Alka Setiya[1] | Rachna Behl[2]

[1] M.Tech Student, Department of Computer Science & Engineering, Faculty of Engineering and Technology, MRIU at Faridabad, Haryana, India.

[2] Associate Professor, Department of Computer Science & Engineering, Faculty of Engineering & Technology, MRIU at Faridabad, Haryana, India.

ABSTRACT

The importance and motivations behind geospatial data extraction has been changing and developing step by step, energized more commitment by the OSM. This development gave progressive techniques for sharing furthermore, processing information by crowd-sourcing, for example, OSM, likewise called "the wikification of maps" by a few analysts. At the point when crowd-sourcing gathers immense information which withhold in corpus data, with help of overall territorial jurisdiction with fluctuating level of mapping background, the concentration of this scheme ought to be on breaking down the information as opposed to gathering it, using semantic parser we provide the scheme to query OSM information by looking at it with restrictive information or information of administrative guide offices and surveys the exploration work for appraisal of OSM and furthermore talks about the future bearings using Machine Readable Language. Therefore, in this scheme we propose to use the corpus data more than 500 MB from maps to extract an accurate semantic structure that will build the basis of a natural language interface to OSM. Furthermore, we use response-based learning on parser results to adapt a statistical machine translation system for relational database access to OSM.

KEY WORDS: OSM (Open Street Maps), geospatial data, machine readable language, fuzzy natural language. statistical machine translation, semantic parsing.

I. INTRODUCTION

OpenStreetMap (OSM) is a community-built database of geographic data, containing user contributed local and up-to-date information about landmarks all over the world. While the main API is optimized for editing map data, there exists an API that allow to filter map data based on search criteria such as location, type of objects, or features with which objects are tagged. However, issuing a query that is executable against the OSM database still requires detailed knowledge of database internals, something that cannot be expected from a layman user. The goal of our work is the development of an interface to OSM that lets a user ask a question in natural language, which is then parsed into a database query that is executable against a web based filtering tool and returns OSM data on an interactive map. To find such information one would have to issue a query that requires detailed knowledge of the database and the query language:
"area[name='Delhi']→.a;node(area.a)
[name='cinema']→.b;node(around.b:1000)
[entertainment='cinema '][wheelchair='yes'];out;". As a starting point for a natural language interface we built a corpus of 2,380 natural language queries paired with machine readable language (MRL) formulae that we used to extract information using a xml parser. We choose to manually creating a corpus of OSMs from which structure and weights of an xml parser can be learned for some reasons. We will present the OSM community with a set of sample questions that can be executed and whose database query representation can be inspected, below depicts the same data of OSM corpus.

```
<?xml version="1.0" encoding="UTF-8"?>
 <osm version="0.6"/>
        <node id="1" lat="21.4219827" lon="39.8336534">
        <tag k="traffic" v="light"/>
        </node>
        <node id="2" lat="21.4221823" lon="39.8331833">
                <tag k="highway" v="motorway_junction"/>
        </node>
        . . .
        <way id="6">
                <nd ref="1"/>
                <nd ref="2"/>
                <tag k="highway" v="service"/>
        </way>
        <way id="8">
                <nd ref="4"/>
                <nd ref="5"/>
        <tag k="type" v="multipolygon"/>
        </way>
        . . .
        <relation id="2">
                <member type="relation" ref="1" role="inner"/>
                <member type="way" ref="6" role="inner"/>
        <tag k="highway" v="primary"/>
        </relation>
        . . .
</osm>
```

Figure 1: Corpus data format

It will help OSM users and developers to see how the complex geographical facts can be issued as simple natural language queries that are parsed into executable filters on OSM objects.

We are presenting the design and implementation of a parsing and interpretation framework for the Extensible Markup Language (XML) and the Resource Description Framework (RDF). These include high flexibility in interpretation of markup and meta-data, on the one hand, and efficient text parsing facilities, on the other. Parsing of XML text does not need to be highly customizable, but it has to be efficient and reliable. Several fast parsers for XML exist which can be reused as off-the-shelf components. We decided to re-use the Expat parser .In the framework presented here, an object-oriented XML parser wraps these third-party parsers.

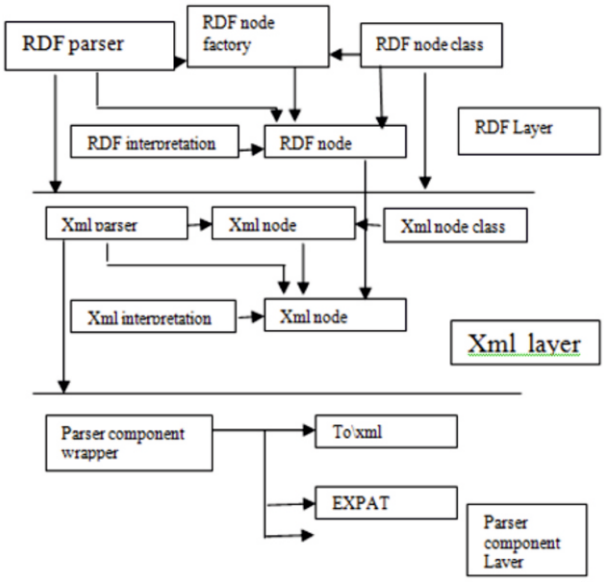

Figure 3: Data Extraction using RDF/XML Parsers

OpenStreetMap whole world dataset is free and accessible as an XML 500 GB file called Planet.osm, updated on weekly basis. Figure 1 gives a snippet of the Planet.osm XML file. The file consists of the following three primitive data types:

(1) Node, which is defined as a point in the space associated with a node identifier, latitude and longitude coordinates,

(2) Way, which represents a line between two nodes, and associated with the way identifier and the two nodes identifiers of the two end points of the line. The line could be simply a road, a mall, a parking, city/country boundary, or part of a lake contour,

(3) Relation, which represents the relation between nodes, ways, or even other relation, and is used to express polygons. For example, to express the boundaries of a certain lake, the nodes need to be defined, then the ways that connect nodes to each other, then a relation that connects the ways together to express the lake boundary.

II. EXISTING WORK & SURVEY

Graham et al.,[1] Slowly but surely OSM is gaining popularity in these countries. Perhaps OpenStreetMap is helping address the participation inequality that is strongly represented in many types of User Generated Content on the Internet today. This "Digital Divide" indicates that very small groups with specific demographic and geographical characteristics are responsible for production of most of the UGC we see on the Internet today. However, these map visualizations indicate that OpenStreetMap is reaching into countries and regions which heretofore would have felt the consequences of the digital divide. Improvements in ICT infrastructures and IT education for socially deprived groups such as women and children coupled with more ubiquitous access to smart phone technology has provided an environment where participation in OpenStreetMap can increase. Research will need to be undertaken to gain a better understanding of the social processes involved in these changes.

Jokar Arsanjani [2], OpenStreetMap has its own geography across time and space. In other words, rarely see identical patterns of contributions in two different regions/countries. When speaking of OpenStreetMap quality and contributions networks the importance of studying diverse case studies has been highlighted. Hence, in this section, two different maps are generated from the OSM statistics, which demonstrate the heterogeneity of OSM in different countries. This map displays a thematic categorization of created nodes, which is one of the key elements in measuring OSM contributions.

Weiwei Sun [3], This paper studies the merged aggregate nearest neighbor(MANN) query. Weiwei sun develop an algorithm for processing this query, the Fast-Pruning algorithm. It uses the Euclidean aggregate distance between a target point and the query set as the pruning distance to prune away unnecessary target points, the experiment results show that it can discard a considerable part of target points which in turn save the execution time and I/O cost., The optimal location query problem based on road networks. Specifically, a road network on which some clients and servers are located. Each client finds the server that is closest to her for service and her cost of getting served is equal to the(network) distance between the client and the server serving her multiplied by her weight or importance. The optimal location query problem is to find a location for setting up a new server such that the maximum cost of clients being served by the servers(including the new server) is minimized. This problem has been studied before, but the state-of-the-art is still not efficient enough. In this paper, author propose an efficient algorithm for the optimal location query problem, which is based on a novel idea of nearest location component. They also discuss three extensions of the optimal location query problem, namely the optimal multiple-location query problem, the optimal location query problem on 3D road networks, and the optimal location query problem with another objective. Extensive experiments were conducted which showed that our algorithms are faster than the state-of-the-art by at least an order of magnitude on large real benchmark datasets. For example, on our largest real datasets, the state-of-the-art ran for more than 10 hours but our algorithm ran within 3 minutes only (i.e., >200 times faster).

Lingkun Wu†, Xiaokui Xia [6], This paper presents an experimental comparison of four state-of-the-art techniques for answering shortest path and distance queries on road networks, namely, SILC, PCPD, CH, and TNR. They used a variety of real datasets with up to twenty million vertices, and evaluated each technique in terms of its preprocessing time, space overhead, and query efficiency. First, CH is the most space economic technique compared with TNR, SILC and PCPD, and yet, it is the second most efficient technique in answering shortest path and distance queries. Therefore, SILC is recommended for processing shortest path queries when time efficiency is crucial and space overhead is less concerned. Finally, although PCPD was proposed as a successor to SILC with an improved asymptotic space complexity, its practical performance (in terms of preprocessing time, space consumption, and query efficiency) is inferior to SILC.

L. Alarabi, A. Eldawy, R. Alghamdi, and M. F. Mokbel. TAREEG [7], This demo presents TAREEG; a web-service that makes real spatial data, from anywhere in the world, available at the fingertips of every researcher or individual. TAREEG gets all its data by leveraging the richness of OpenStreetMap dataset; the most comprehensive available spatial data of the world. Yet, it is still challenging to obtain OpenStreetMap data due to the size limitations, special data format, and the noisy nature of spatial data. TAREEG employs MapReduce-based techniques to make it efficient and easy to extract OpenStreetMap data in a standard form with minimal effort.

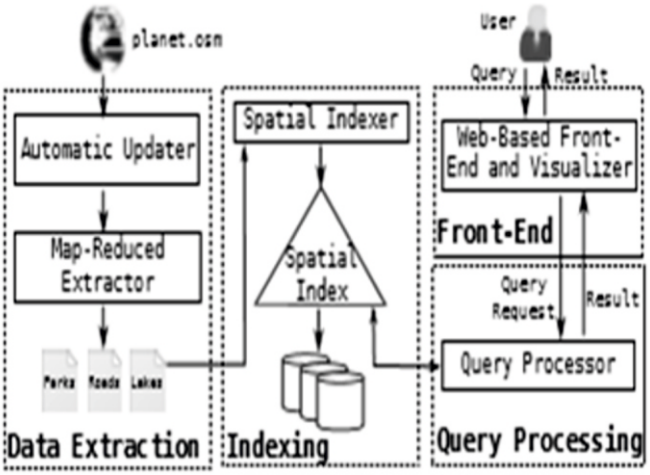

Figure 4: Semantic Parsers

K. Deng, X. Zhou [10], In this paper, three novel algorithms have been proposed for processing multi-source relative skyline queries in road networks. It is not only the first effort to process relative skyline queries in road networks, but also the first study on skyline queries by considering relative network distances to multiple query points at the same time. This experiments confirmed that LBC has the best performance consistently for various test settings. The path distance lower bound approach, based on which LBC is designed, can be applied to benefit other types of road network queries where network distance comparison is needed.

A. Guttman. R-Trees [14], The R-tree structure has been shown to be useful for indexing spatial data objects that have non-zero size Nodes corresponding to disk pages of reasonable size (e g 1024 bytes) have values of A4 that produce good performance With smaller nodes the structure should also be effective as a main-memory index, CPU performance would be comparable but there would be no I/O cost. The linear node-split algorithm proved to be as good as more expensive techniques It was fast, and the slightly worse quality of the splits did not affect search performance.

III. PROPOSED EXPEREMENTAL WORK

OSM is introduced as a new knowledge base that has not, to the best of our knowledge, been used for question answering, and offer a new corpus to the research community. Work builds the basis of a natural language interface to OSM that will be enabling for interesting directions of future research, e.g., response-based learning to improve parsing and multilingual database access.

1. **Data Extraction:** Extracting data from OpenStreetMap isn't a trivial task. The entire OpenStreetMap dataset is kept consecutive in one giant volume go into a semi-structured XML format. The Data Extraction module take the uniform resource locator of the compressed Planet.osm file as associate degree input, and outputs many classified files. every computer file contains a consistent spatial set of data, e.g., cinema hall detail & parking facility. we will discuss every extraction job individually based on arc extraction, below algorithm depicts the modus operandi for augmentation.

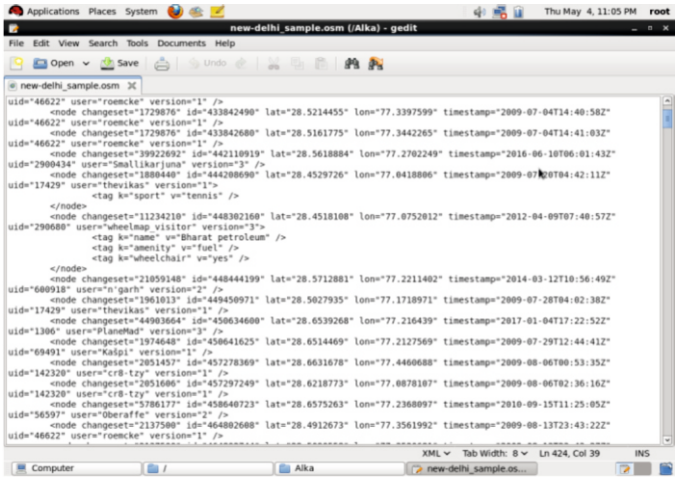

Figure 5: Extracted data

2. **Parsing:** Our corpus creation process was guided by the goal to pair a diverse range of questions with machine readable language (MRL) formulae. These should include the most important OSM tags so that the parser is able to learn a mapping between these tags and the different corresponding

natural language expressions. . It is unambiguously defined via a context-free grammar (CFG) so that one can always ascertain whether or not a formula is valid.

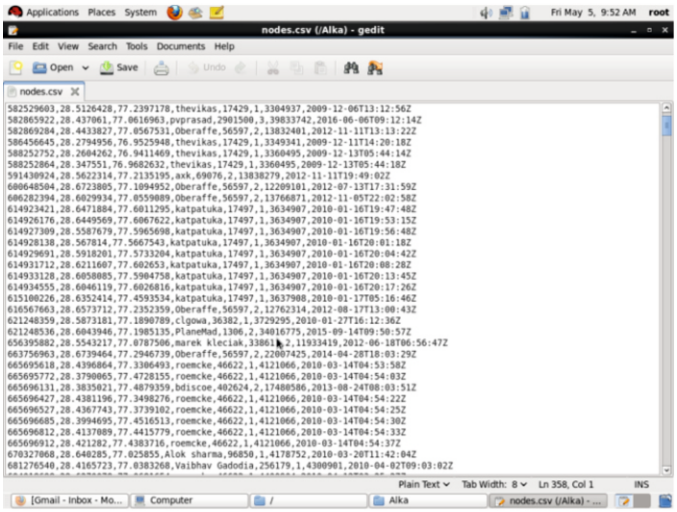

Figure 6: Data in form of Nodes & Node_tags

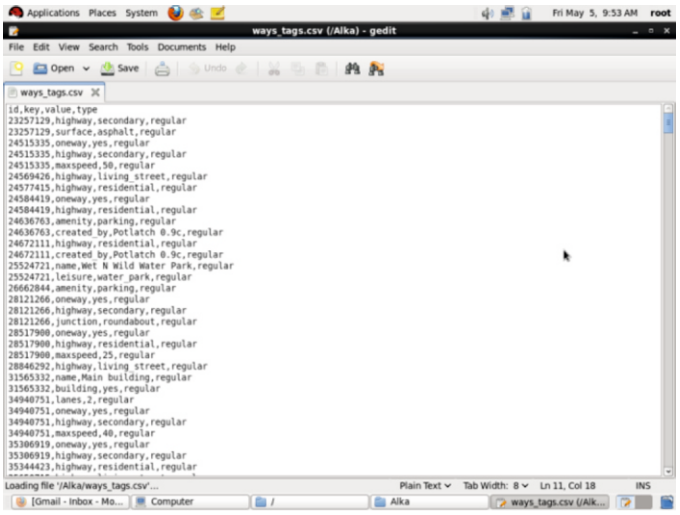

Figure 7: Data in form of Tags, Ways & Ways_tags

3. **Query Creation:** Since even the Overpass Query Wizard requires users to be familiar with the tag set of OSM key-value pairs, it unusable for users with only casual or no knowledge of OSM's internal structure. Nonetheless, we could use parts of the user query log to formulate natural language questions. For example users would enter the query "Which mall in delhi have wheelchair access ?" with MRL "query(area(keyval(name,' Delhi')),nwr(keyval(entertainment,' mall'),keyval(wheelchair,'yes')), findkey(name))" can be presented as a tree. A preorder traversal gives: "query@3 area@1 keyval@2 name@0 delhi@s nwr@2 keyval@2 tourism@0 mall@s keyval@2 wheelchair@0 yes@s findkey@1 name@0"

4. **Query Operator:** A single database query is encoded in the operator query() which will hold the Overpass query as well as further specifications about what kind of answer should be retrieved. A few operators are directly derived from Overpass, merely re-written as a tree structure. As such OSM key-value pairs are encoded using the operator keyval() which takes 2 arguments, the first being the key and the second the value. The area operator from Overpass directly translates to the operator area(). Nodes, ways and relations are grouped together under the nwr() operator which will supply the union of the query run with the 3 types in turn. This is necessary because often buildings, e.g. schools, may be represented as any of the 3 types depending on how specific the annotator wanted to be. Both area() and nwr() then take one ore more keyval() arguments.

5. **Other Operators:** Some further operators were needed to model the MRL formula for complex questions. and() is used when the user asks for two different nuggets of information ("Where is the nearest mall and the closest parking?" or "Give me the website and name of ..."). or() is used to create unions, as for example, needed in a sentence such as "Give me the closest mall or restaurant." "*" can be used as a wild card in a value position, e.g. ['parking'=*] will returned any historic objects, be it a castle, a monument or something else. nodup() returns a set with no duplicates.

Algorithm with multi-polygon augmentation semantic parsing algorithm :

Input : Split S

Output: Position of last byte Key , Primitive OSM element Value

1 Download maps from OpenStreetMap

2 Fetching the corpus data

3 Parsing corpus data using EXPAT parsing

4 Forming tags or structure layout

5 Apply K-Means Algorithm

5 Mignating & auditioning semantics

6 Transforming corpus data to Comma Separated values

7 Integrated with relational database

12 Generate query plan

13 Results based on queries

14 Termination;

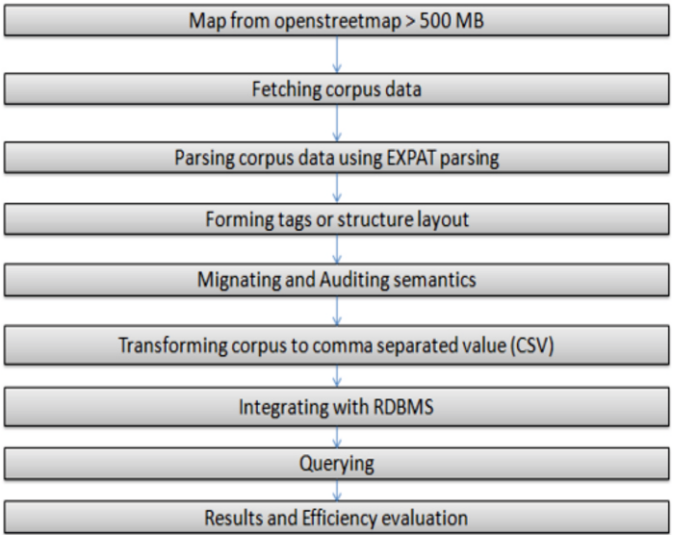

Figure 8: Flow Diagram

In the scheme, we identify the polygon augmentation expat parsing algorithm with the help to extract corpus extraction. The scheme will provide 90% to 95% query based accurate information with much efficient time then spatial index retrieval system.

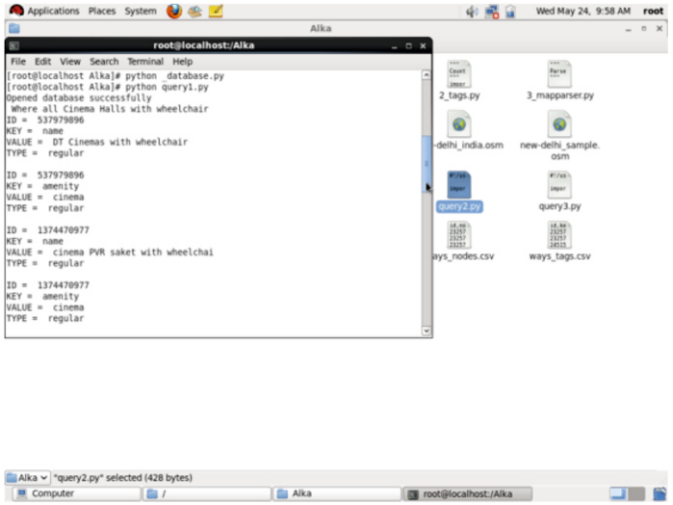

Figure:9 Query1 (Cinema hall providing the wheelchair facility)

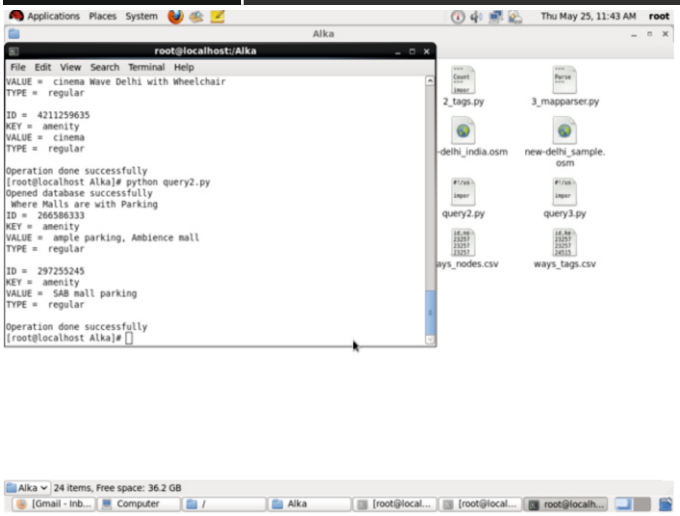

Figure: 10 Query2(Mall providing the parking facility)

CONCLUSION

We will develop an approach to query the OSM database for complex geographical facts via natural language questions. The key technology is a semantic parser that is trained in supervised fashion from a large set of questions annotated with executable MRLs. Our corpus is larger than previous annotated question-answer corpora, while including a wide variety of challenging questions. Terms such as "nearby", "in the south of'", "within x miles" are particularly well-suited for a natural language query interface.

REFERENCES

[1] Graham, M.; Hale, S.; Stephens, M. Digital Divide: "The Geography of Internet Access. Environ. Plan". A 2012, 44, 1009–1010.

[2] Jokar Arsanjani, J.; Helbich, M.; Bakillah, M.; Loos, L. "The Emergence and Evolution of OpenStreetMap: A Cellular Automata Approach". Int. J. Digit. Earth 2013 b, 00, 1–15

[3] Weiwei Sun, Chong Chen, "Merged aggregate nearest neighbor query processing in road network", Singapore Management University Institutional Knowledge at Singapore Management University, 10-2013.

[4] Zitong Chen, Yubao Liu, Raymond Chi-Wing Wong, "Efficient Algorithms for Optimal Location Queries in Road Networks", The Hong Kong University of Science and Technology, Hong Kong, China.

[5] Aye Su Yee Win, "Fast Algorithm for Multi-type Nearest Neighbor Quer", Graduate School of Science and Engineering, Saitama University, D-002.

[6] Lingkun Wu, Xiaokui Xiao, "Shortest Path and Distance Queries on Road Networks: An Experimental Evaluation", School of Computer Engineering , Nanyang Technological University, Singapore.

[7] L. Alarabi, A. Eldawy, R. Alghamdi, and M. F. Mokbel. TAREEG: A MapReduce-Based Web Service for Extracting Spatial Data from OpenStreetMap (System Demonstration). In SIGMOD, pages 897–900, Snowbird, UT, June 2014.

[8 Z. Chen, Y. Liy, R. C.-W. Wong, J. Xiong, G. Mai, and C. Long. "Efficient algorithms for optimal location queries in road networks". In SIGMOD, 2014.

[9] Z. Chen, H. T. Shen, X. Zhou, and J. X. Yu. Monitoring path nearest neighbor in road networks. In SIGMOD, pages 591–602, 2009.

[10] K. Deng, X. Zhou, and H. T. Shen. "Multi-source skyline query processing in road networks". In ICDE, pages 796–805, 2007.

[11] A. Eldawy and M. F. Mokbel. A Demonstration of SpatialHadoop: An Efficient MapReduce Framework for Spatial Data (System Demo). In VLDB, Riva del Garda, Italy, Aug. 2013.

[12] Z. Chen, Y. Liy, R. C.-W. Wong, J. Xiong, G. Mai, and C. Long. "Efficient algorithms for optimal location queries in road networks"s. In SIGMOD, 2014.

[13] Z. Chen, H. T. Shen, X. Zhou, and J. X. Yu. Monitoring path nearest neighbor in road networks. In SIGMOD, pages 591–602, 2009.

[14] K. Deng, X. Zhou, and H. T. Shen. "Multi-source skyline query processing in road networks". In ICDE, pages 796–805, 2007.

[15] A. Eldawy and M. F. Mokbel. A Demonstration of SpatialHadoop: An Efficient MapReduce Framework for Spatial Data (System Demo). In VLDB, Riva del Garda, Italy, Aug. 2013.